UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/789,311	02/27/2004	Sheueling Chang Shantz	6000-31500	9201

58467 7590 07/22/2010

MHKKG/Oracle (Sun)
P.O. BOX 398
AUSTIN, TX 78767

EXAMINER	
JOHNSON, CARLTON	

ART UNIT	PAPER NUMBER
2436	

NOTIFICATION DATE	DELIVERY MODE
07/22/2010	ELECTRONIC

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Notice of the Office communication was sent electronically on above-indicated "Notification Date" to the following e-mail address(es):

patent_docketing@intprop.com
ptomhkkg@gmail.com

Office Action Summary	Application No.	Applicant(s)	
	10/789,311	SHANTZ ET AL.	
	Examiner	Art Unit	
	CARLTON V. JOHNSON	2436	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE *3* MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *04 May 2010*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-67* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-67* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☐ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

1. This action is in response to application amendments filed on 5-4-2010.

2. Claims **1 - 67** are pending. Claims **1, 21, 38, 53, 66, 67** are independent. This application was filed on 2-27-2004.

Response to Arguments

3. Applicant's arguments have been fully considered but they were not persuasive.

3.1 Applicant argues that the referenced prior art does not disclose, *add-chaining operation and multiply-chaining operation*.

The add-chaining operation and multiply-chaining operation discloses a combination (chaining) of accumulate and multiplication operations into a single arithmetic instruction. The Specification in paragraph [1076] states that the instruction performs multiply-accumulate-chaining operation, which combines (appears to be equivalent to prior art chaining) add-chaining and multiply-chaining in one operation in order to avoid the multiplier latency from the processing of operand(s) information between arithmetic operations. This statement appears to suggest the combination or chaining of two or more arithmetic operations with an implicit input or transfer of operand(s) between arithmetic instructions. Contrary to Applicant's assertions, the referenced prior art appears to disclose an equivalent combined arithmetic instruction as the claimed invention.

The concept of the claimed invention is to combine a set of arithmetic operations (accumulate and multiplication arithmetic operations) into a single arithmetic instruction or invocation. An arithmetic operation generates a resultant operand which is

transferred between two (successive) arithmetic operations. The transfer of this generated operand is the feedback mechanism indicated by the claimed invention. The resultant or feedback operand is implicitly (without an explicit operation) transferred to the next arithmetic operation or instruction in the sequence. Applicant's invention discloses that the resultant operand can be a partial result (high-order or low-order bits from a resultant word.

Huppenthal discloses an architecture for chaining a number of arithmetic operations (such as multiplication, accumulation, and etc) to form a single arithmetic instruction. The single arithmetic instruction is initiated and the sequence of chained operations is performed with operand transfer controlled by the computing system architecture via the usage of a chain port mechanism. The chain port mechanism supplies operands to each successive arithmetic operation in the sequence. The chain port mechanism supplies operands without any support from the processor. Hinds discloses the generation of a partial result (high-order or low-order bits). Huppenthal and Hinds disclose chaining a set of two or more arithmetic operations within a single arithmetic instruction and the generation of a partial result as the operand that is implicitly transferred between the arithmetic instructions.

3.2 Applicant argues that the referenced prior art does not disclose, *add-chaining and multiply-chaining as part of an instruction set architecture.*

Huppenthal and Hinds disclose the concept of chaining of arithmetic operations. This chaining concept is part of the architecture of the computing system disclosed within the

Huppenthal prior art claimed invention. The chain operation is a legitimate operation within the instruction set of the Huppenthal prior art.

3.3 Applicant argues that the referenced prior art does not disclose, *obviousness rejection.*

A 103 rejection based on multiple references is a legitimate technique according to the MPEP. The current application is rejected based on the Huppenthal, Hinds and Chen prior art references. The set of references are in a same field of endeavor as the claimed invention, computing system instruction set processing. The 103 rejection allows portions of a claimed invention to come from different prior art references. The prior art references are within the same field of high performance computations which can be used for cryptographic operations such as encryption key generation.

3.4 Applicant argues that the referenced prior art does not disclose, *Independent Claims 21, 38, 53, 66, 67.*

Independent claims 21, 38, 53, 66, 67 have similar limitations as independent claim 1. Reponses to arguments for independent claim 1 respond to arguments against independent claims 21, 38, 53, 66, 67.

Claim Rejections - 35 USC § 103

4. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

5. Claims **1, 4 - 10, 19, 21 - 26, 36, 38 - 42, 48, 52 - 60, 62, 66, 67** are rejected under 35 U.S.C. 103(a) as being unpatentable over **Huppenthal et al.** (US Patent No. **6,339,819**) in view of **Hinds et al.** (US Patent No. **6,542,916**) and further in view of **Chen et al.** (US Patent No. **6,763,365**).

With Regards to Claim 1, Huppenthal discloses a method implemented in a device and storing the first partial result; and using the stored first partial result in a subsequent computation in the public-key cryptography application, the method comprising.
a previously executed single arithmetic instruction of a processor set and wherein the currently executing single arithmetic instructions does not include an explicit source operand for specifying the high order bits. (Huppenthal col. 3, lines 1-7: number of MAP elements chained together to accomplish a single function or operation (implies a single arithmetic instruction; col. 3, lines 18-25: MAP elements can receive operands via chained port; col. 18, line 66 - col. 19, line 3: output data from one MAP element to be sent directly to the user array of the next MAP element with no processor intervention via a chain port)
Huppenthal does not specifically disclose a multiplying and summing sequence. However, Hinds discloses a first arithmetic circuit comprising a first plurality of arithmetic structures feeding back high order bits of an executed arithmetic instruction in the public-key cryptography application, generated by the first arithmetic circuit, to a second arithmetic circuit comprising a second plurality of arithmetic structures and the second arithmetic circuit, generating a first partial result of a currently executing arithmetic

instruction in the public-key cryptography application, the first partial result representing the high order bits summed with low order bits of a result of a first number multiplied by a second number, the summing of the high order bits being performed during multiplication of the first number and the second number, the summing and at least a portion of the multiplication being performed in the second arithmetic circuit. (Hines col. 3, lines 5-17: applying a multiply-accumulate operation to operands; multiplying operands and adding multiplication results; col. 8, lines col. 8, lines 6-25: chained multiply-accumulate operation; carry save adders and usage of partial multiplier (partial results: high order bits); output of results of partial multiplier is normalized and passed to carry save adders and final product adder)

Huppenthal-Hinds does not specifically disclose supporting a cryptographic application. However, Chen discloses supporting a public-key cryptography application. (see Chen col. 6, lines 23-25: arithmetic operations to support acceleration of cryptographic functions)

It would have been obvious to one of ordinary skill in the art to modify Huppenthal-Hinds to support a cryptographic application as taught by Chen. One of ordinary skill in the art would have been motivated to employ the teachings of Chen to greatly improve the performance of cryptographic circuits. (Chen col. 5, lines 40-42)

With Regards to Claim 4, Huppenthal discloses the method as recited in claim 1. further comprising feeding back the high order bits through a register to the second arithmetic circuit. (Huppenthal col. 3, lines 1-7: number of MAP elements chained

together to accomplish a single function or operation (implies a single arithmetic instruction; col. 3, lines 18-25: MAP elements can receive operands via chained port; col. 18, line 66 - col. 19, line 3: output data from one MAP element to be sent directly to the user array of the next MAP element with no processor intervention via a chain port)

With Regards to Claim 5, Huppenthal discloses the method as recited in claim 1. Huppenthal does not specifically disclose supporting a cryptographic application. However, Hinds discloses generating a second partial result of the currently executing single arithmetic instruction in the first arithmetic circuit, the second partial result representing the high order bits of the multiplication result of the first number multiplied by the second number. (Hines col. 3, lines 5-17: applying a multiply-accumulate operation to operands; multiplying operands and adding multiplication results; col. 8, lines col. 8, lines 6-25: chained multiply-accumulate operation; carry save adders and usage of partial multiplier (partial results: high order bits); output of results of partial multiplier is normalized and passed to carry save adders and final product adder) Motivation for Hinds to disclose arithmetic operations is as stated in Claim 1 above.

With Regards to Claim 6, Huppenthal discloses the method as recited in claim 1. Huppenthal does not specifically disclose a multiplying and summing sequence. However, Hinds discloses generating a second partial result of the currently executing single arithmetic instruction, the second partial result representing the high order bits of the multiplication result of the first number multiplied by the second number summed

with the high order bits of the executed arithmetic instruction previously executed single arithmetic instruction. (Hines col. 3, lines 5-17: applying a multiply-accumulate operation to operands; multiplying operands and adding multiplication results; col. 8, lines col. 8, lines 6-25: chained multiply-accumulate operation; carry save adders and usage of partial multiplier (partial results: high order bits); output of results of partial multiplier is normalized and passed to carry save adders and final product adder)
Motivation for Hinds to disclose arithmetic operations is as stated in Claim 1 above.

With Regards to Claim 7, Huppenthal discloses the method as recited in claim 6. However, Hinds discloses supplying values generated in one or more most significant columns of the second arithmetic structures to one or more least significant columns of the first arithmetic structures while generating the first and second partial results. (Hines col. 3, lines 5-17: applying a multiply-accumulate operation to operands; multiplying operands and adding multiplication results; col. 8, lines col. 8, lines 6-25: chained multiply-accumulate operation; carry save adders and usage of partial multiplier (partial results: high order bits); output of results of partial multiplier is normalized and passed to carry save adders and final product adder)
Motivation for Hinds to disclose arithmetic operations is as stated in Claim 1 above.

With Regards to Claim 8, Huppenthal discloses the method as recited in claim 5. Huppenthal does not specifically disclose a multiplying and summing sequence. However, Hinds discloses generating of the first and second partial result is in response

to execution of a currently executing single arithmetic instruction. (Hines col. 3, lines 5-17: applying a multiply-accumulate operation to operands; multiplying operands and adding multiplication results; col. 8, lines col. 8, lines 6-25: chained multiply-accumulate operation; carry save adders and usage of partial multiplier (partial results: high order bits); output of results of partial multiplier is normalized and passed to carry save adders and final product adder)

Motivation for Hinds to disclose arithmetic operations is as stated in Claim 1 above.

With Regards to Claim 9, Huppenthal discloses the method as recited in claim 6. Huppenthal does not specifically disclose a multiplying and summing sequence. However, Hinds discloses generating of the first and second partial result is in response to execution of a currently executing single arithmetic instruction.

Motivation for Hinds to disclose arithmetic operations is as stated in Claim 1 above.

With Regards to Claim 10, Huppenthal discloses the method as recited in claim 1. Huppenthal does not specifically disclose a multiplying and summing sequence. However, Hinds discloses at least one of the first and second pluralities of arithmetic structures comprises a plurality of carry save adder tree columns. (Hines col. 3, lines 5-17: applying a multiply-accumulate operation to operands; multiplying operands and adding multiplication results; col. 8, lines col. 8, lines 6-25: chained multiply-accumulate operation; carry save adders and usage of partial multiplier (partial results: high order bits); output of results of partial multiplier is normalized and passed to carry save adders

and final product adder)

Motivation for Hinds to disclose arithmetic operations is as stated in Claim 1 above.

With Regards to Claim 19, Huppenthal discloses the method as recited in claim 1. Huppenthal does not specifically disclose a multiplying and summing sequence. However, Hinds discloses feeding back high order bits of the currently executing arithmetic instruction from the first arithmetic circuit to the second arithmetic circuit for use with execution of a subsequent single arithmetic instruction. (Hines col. 3, lines 5-17: applying a multiply-accumulate operation to operands; multiplying operands and adding multiplication results; col. 8, lines col. 8, lines 6-25: chained multiply-accumulate operation; carry save adders and usage of partial multiplier (partial results: high order bits); output of results of partial multiplier is normalized and passed to carry save adders and final product adder)

Motivation for Hinds to disclose arithmetic operations is as stated in Claim 1 above.

With Regards to Claim 21, Huppenthal discloses a method implemented in a device and storing the first partial result; and using the first partial result in a subsequent computation wherein the currently executing single arithmetic instruction does not include an explicit source operand for specifying the high order bits. (Huppenthal col. 3, lines 1-7: number of MAP elements chained together to accomplish a single function or operation (implies a single arithmetic instruction; col. 3, lines 18-25: MAP elements can receive operands via chained port; col. 18, line 66 - col. 19, line 3: output data from one

MAP element to be sent directly to the user array of the next MAP element with no processor intervention via a chain port)

Huppenthal does not specifically disclose a multiplying and summing sequence. However, Hinds discloses a first arithmetic circuit comprising a first plurality of arithmetic structures feeding back high order bits of an executed single arithmetic instruction of a processor instruction set in the public-key cryptography application, generated by the first arithmetic circuit to a second arithmetic circuit comprising a second plurality of arithmetic structures; supplying a third number to the second arithmetic circuit; the second arithmetic circuit generating a first partial result of a currently executing single arithmetic instruction in the public-key cryptography application, the first partial result being a representation of the high order bits summed with, low order bits of a result of a first number multiplied by a second number, and with the third number, the summing being performed during multiplication of the first number and the second number, the summing and at least a portion of the multiplication being performed in the second arithmetic circuit. (Hines col. 3, lines 5-17: applying a multiply-accumulate operation to operands; multiplying operands and adding multiplication results; col. 8, lines col. 8, lines 6-25: chained multiply-accumulate operation; carry save adders and usage of partial multiplier (partial results: high order bits); output of results of partial multiplier is normalized and passed to carry save adders and final product adder)

Motivation for Hinds to disclose arithmetic operations is as stated in Claim 1 above.

Huppenthal-Hinds does not specifically disclose supporting a cryptographic application. However, Chen discloses supporting public-key cryptography application (see Chen col.

6, lines 23-25: arithmetic operations to support acceleration of cryptographic functions)
Motivation for Chen to disclose supporting public-key cryptography application is as stated in Claim 1 above.

With Regards to Claim 22, Huppenthal discloses the method as recited in claim 21. Huppenthal does not specifically disclose a multiplying and summing sequence. However, Hinds discloses feeding back the high order bits through a register to the second arithmetic circuit. (Hines col. 3, lines 5-17: applying a multiply-accumulate operation to operands; multiplying operands and adding multiplication results; col. 8, lines col. 8, lines 6-25: chained multiply-accumulate operation; carry save adders and usage of partial multiplier (partial results: high order bits); output of results of partial multiplier is normalized and passed to carry save adders and final product adder)
Motivation for Hinds to disclose arithmetic operations is as stated in Claim 1 above.

With Regards to Claim 23, Huppenthal discloses the method as recited in claim 21. Huppenthal does not specifically disclose a multiplying and summing sequence. However, Hinds discloses the first arithmetic circuit generating a second partial result of the currently executing arithmetic instruction, the second partial result representing the high order bits of the multiplication result of the first number multiplied by the second number.
Motivation for Hinds to disclose arithmetic operations is as stated in Claim 1 above.

With Regards to Claim 24, Huppenthal discloses the method as recited in claim 21. Huppenthal does not specifically disclose a multiplying and summing sequence. However, Hinds discloses generating a second partial result of the currently executing arithmetic instruction, the second partial result representing the high order bits of the multiplication result of the first number multiplied by the second number summed with the high order bits of the previously executed arithmetic instruction and the third number. (Hines col. 3, lines 5-17: applying a multiply-accumulate operation to operands; multiplying operands and adding multiplication results; col. 8, lines col. 8, lines 6-25: chained multiply-accumulate operation; carry save adders and usage of partial multiplier (partial results: high order bits); output of results of partial multiplier is normalized and passed to carry save adders and final product adder)
Motivation for Hinds to disclose arithmetic operations is as stated in Claim 1 above.

With Regards to Claim 25, Huppenthal discloses the method as recited in claim 24. Huppenthal does not specifically disclose a multiplying and summing sequence. However, Hinds discloses supplying values generated in one or more most significant columns of the second arithmetic structures to one or more least significant columns of the first arithmetic structures while generating the first and second partial results. (Hines col. 3, lines 5-17: applying a multiply-accumulate operation to operands; multiplying operands and adding multiplication results; col. 8, lines col. 8, lines 6-25: chained multiply-accumulate operation; carry save adders and usage of partial multiplier (partial results: high order bits); output of results of partial multiplier is normalized and passed to

carry save adders and final product adder)

Motivation for Hinds to disclose arithmetic operations is as stated in Claim 1 above.

With Regards to Claim 26, Huppenthal discloses the method as recited in claim 23. Huppenthal does not specifically disclose a multiplying and summing sequence. However, Hinds discloses generating of the first and second partial result is in response to execution of a single arithmetic instruction. (Hines col. 3, lines 5-17: applying a multiply-accumulate operation to operands; multiplying operands and adding multiplication results; col. 8, lines col. 8, lines 6-25: chained multiply-accumulate operation; carry save adders and usage of partial multiplier (partial results: high order bits); output of results of partial multiplier is normalized and passed to carry save adders and final product adder)

Motivation for Hinds to disclose arithmetic operations is as stated in Claim 1 above.

With Regards to Claim 36, Huppenthal discloses the method as recited in claim 21 further comprising feeding back high order bits of the currently executing arithmetic instruction from the first arithmetic circuit to the second arithmetic circuit for use with execution of a subsequent single arithmetic instruction. (Huppenthal col. 3, lines 1-7: number of MAP elements chained together to accomplish a single function or operation (implies a single arithmetic instruction; col. 3, lines 18-25: MAP elements can receive operands via chained port; col. 18, line 66 - col. 19, line 3: output data from one MAP element to be sent directly to the user array of the next MAP element with no processor

intervention via a chain port)

With Regards to Claim 38, Huppenthal discloses a processor. (Huppenthal col. 4, lines 43-47: processor; col. 3, lines 1-7: number of MAP elements chained together to accomplish a single function or operation (implies a single arithmetic instruction; col. 3, lines 18-25: MAP elements can receive operands via chained port; col. 18, line 66 - col. 19, line 3: output data from one MAP element to be sent directly to the user array of the next MAP element with no processor intervention via a chain port),
Huppenthal does not specifically disclose a multiplying and summing sequence. However, Hinds discloses a first plurality of arithmetic structures configured to generate high order bits for an arithmetic operation in a public-key cryptography application that includes a multiplication operation; and a second plurality of arithmetic structures configured to generate low order bits of the arithmetic operation; wherein the second arithmetic structures are further configured to receive the high order bits generated by the first plurality of arithmetic structures during a previous arithmetic operation in the public-key cryptography application and to generate a first partial result of the arithmetic operation, the first partial result representing the high order bits summed with low order bits of a multiplication result of the multiplication operation; and wherein the processor further comprises a register configured to store the first partial result for use in a subsequent arithmetic operation in the public-key cryptography application. (Hines col. 3, lines 5-17: applying a multiply-accumulate operation to operands; multiplying operands and adding multiplication results; col. 8, lines col. 8, lines 6-25: chained

multiply-accumulate operation; carry save adders and usage of partial multiplier (partial results: high order bits); output of results of partial multiplier is normalized and passed to carry save adders and final product adder)

Motivation for Hinds to disclose arithmetic operations is as stated in Claim 1 above.

Huppenthal-Hinds does not specifically disclose supporting a cryptographic application. However, Chen discloses configured to support public-key cryptography applications (see Chen col. 6, lines 23-25: arithmetic operations to support acceleration of cryptographic functions)

Motivation for Chen to disclose supporting a cryptographic application is as stated in Claim 1 above.

With Regards to Claim 39, Huppenthal discloses the processor as recited in claim 38. Huppenthal does not specifically disclose a multiplying and summing sequence. However, Hinds discloses the first arithmetic structures are configured to generate a second partial result of the arithmetic instruction, the second partial result representing the high order bits of the arithmetic operation. (Hines col. 3, lines 5-17: applying a multiply-accumulate operation to operands; multiplying operands and adding multiplication results; col. 8, lines col. 8, lines 6-25: chained multiply-accumulate operation; carry save adders and usage of partial multiplier (partial results: high order bits); output of results of partial multiplier is normalized and passed to carry save adders and final product adder)

Motivation for Hinds to disclose arithmetic operations is as stated in Claim 1 above.

With Regards to Claim 40, Huppenthal discloses the processor as recited in claim 39. Huppenthal does not specifically disclose a multiplying and summing sequence. However, Hinds discloses the second arithmetic structures are further configured to supply values generated in one or more most significant columns of the second arithmetic structures to one or more least significant columns of the first arithmetic structures while generating the first and second partial results. (Hines col. 3, lines 5-17: applying a multiply-accumulate operation to operands; multiplying operands and adding multiplication results; col. 8, lines col. 8, lines 6-25: chained multiply-accumulate operation; carry save adders and usage of partial multiplier (partial results: high order bits); output of results of partial multiplier is normalized and passed to carry save adders and final product adder)
Motivation for Hinds to disclose arithmetic operations is as stated in Claim 1 above.

With Regards to Claim 41, Huppenthal discloses the processor as recited in claim 39, wherein the first and second arithmetic structures are configured to generate of the first and second partial results in response to execution of a single arithmetic instruction. Motivation for Hinds to disclose arithmetic operations is as stated in Claim 1 above.

With Regards to Claim 42, Huppenthal discloses the processor as recited in claim 38, further comprising a register coupled to the first and second arithmetic structures to

supply the high order bits to the second arithmetic structures. (Huppenthal col. 3, lines 1-7: number of MAP elements chained together to accomplish a single function or operation (implies a single arithmetic instruction; col. 3, lines 18-25: MAP elements can receive operands via chained port; col. 18, line 66 - col. 19, line 3: output data from one MAP element to be sent directly to the user array of the next MAP element with no processor intervention via a chain port)

With Regards to Claim 48, Huppenthal discloses the processor as recited in claim 38. Huppenthal does not specifically disclose a multiplying and summing sequence. However, Hinds discloses at least one of the first and second pluralities of arithmetic structures comprises a plurality of carry save adder tree columns. (Hines col. 3, lines 5-17: applying a multiply-accumulate operation to operands; multiplying operands and adding multiplication results; col. 8, lines col. 8, lines 6-25: chained multiply-accumulate operation; carry save adders and usage of partial multiplier (partial results: high order bits); output of results of partial multiplier is normalized and passed to carry save adders and final product adder)
Motivation for Hinds to disclose arithmetic operations is as stated in Claim 1 above.

With Regards to Claim 52, Huppenthal discloses the processor as recited in claim 38, wherein the processor is a general purpose processor. (see Huppenthal col. 4, lines 43-47: processor)

With Regards to Claim 53, Huppenthal discloses a processor and a register configured to store the first partial result for use in a subsequent arithmetic operation in the public-key cryptography application. (Huppenthal col. 4, lines 43-47: processor; col. 3, lines 1-7: number of MAP elements chained together to accomplish a single function or operation (implies a single arithmetic instruction; col. 3, lines 18-25: MAP elements can receive operands via chained port; col. 18, line 66 - col. 19, line 3: output data from one MAP element to be sent directly to the user array of the next MAP element with no processor intervention via a chain port)

Huppenthal does not specifically disclose a multiplying and summing sequence. However, Hinds discloses a first plurality of arithmetic structures configured to generate high order bits for an arithmetic operation in a public-key cryptography application that includes a multiplication operation of a first and a second number; a second plurality of arithmetic structures configured to generate low order bits of the arithmetic operation; wherein the second arithmetic structures are configured to: receive the high order bits generated by the first plurality of arithmetic structures during a previous arithmetic operation; receive a third number; and generate a first partial result of the arithmetic operation, the first partial result representing the high order bits summed with low order bits of a multiplication result of the multiplication operation, and with the third number; and wherein the processor. (Hines col. 3, lines 5-17: applying a multiply-accumulate operation to operands; multiplying operands and adding multiplication results; col. 8, lines col. 8, lines 6-25: chained multiply-accumulate operation; carry save adders and usage of partial multiplier (partial results: high order bits); output of results of partial

multiplier is normalized and passed to carry save adders and final product adder)

Motivation for Hinds to disclose arithmetic operations is as stated in Claim 1 above.

Huppenthal does not specifically disclose supporting public-key cryptography applications.

However, Chen discloses configured to support public-key cryptography applications (see Chen col. 6, lines 23-25: arithmetic operations to support acceleration of cryptographic functions)

Motivation for Chen to disclose supporting public-key cryptography applications is as stated in Claim 1 above.

With Regards to Claim 54, Huppenthal discloses the processor as recited in claim 53.

Huppenthal does not specifically disclose a multiplying and summing sequence.

However, Hinds discloses the first arithmetic structures are further configured to generate a second partial result of the arithmetic instruction, the second partial result representing the high order bits of the arithmetic operation. (Hines col. 3, lines 5-17: applying a multiply-accumulate operation to operands; multiplying operands and adding multiplication results; col. 8, lines col. 8, lines 6-25: chained multiply-accumulate operation; carry save adders and usage of partial multiplier (partial results: high order bits); output of results of partial multiplier is normalized and passed to carry save adders and final product adder)

Motivation for Hinds to disclose arithmetic operations is as stated in Claim 1 above.

With Regards to Claim 55, Huppenthal discloses the processor as recited in claim 54. Huppenthal does not specifically disclose a multiplying and summing sequence. However, Hinds discloses the second arithmetic structures are further configured to generate values in one or more most significant columns and to supply them to one or more least significant columns of the first arithmetic structures while generating the first and second partial results. (Hines col. 3, lines 5-17: applying a multiply-accumulate operation to operands; multiplying operands and adding multiplication results; col. 8, lines col. 8, lines 6-25: chained multiply-accumulate operation; carry save adders and usage of partial multiplier (partial results: high order bits); output of results of partial multiplier is normalized and passed to carry save adders and final product adder) Motivation for Hinds to disclose arithmetic operations is as stated in Claim 1 above.

With Regards to Claim 56, Huppenthal discloses the processor as recited in claim 54. Huppenthal does not specifically disclose a multiplying and summing sequence. However, Hinds discloses the first arithmetic structures are configured to generate of the first and second partial result in response to execution of a single arithmetic instruction. (Hines col. 3, lines 5-17: applying a multiply-accumulate operation to operands; multiplying operands and adding multiplication results; col. 8, lines col. 8, lines 6-25: chained multiply-accumulate operation; carry save adders and usage of partial multiplier (partial results: high order bits); output of results of partial multiplier is normalized and passed to carry save adders and final product adder)
Motivation for Hinds to disclose arithmetic operations is as stated in Claim 1 above.

With Regards to Claim 57, Huppenthal discloses the processor as recited in claim 53. Huppenthal does not specifically disclose a multiplying and summing sequence. However, Hinds discloses a register coupled to the first and second arithmetic structures to supply the high order bits to the second arithmetic structures. (Hines col. 3, lines 5-17: applying a multiply-accumulate operation to operands; multiplying operands and adding multiplication results; col. 8, lines col. 8, lines 6-25: chained multiply-accumulate operation; carry save adders and usage of partial multiplier (partial results: high order bits); output of results of partial multiplier is normalized and passed to carry save adders and final product adder)
Motivation for Hinds to disclose arithmetic operations is as stated in Claim 1 above.

With Regards to Claim 58, Huppenthal discloses the processor as recited in claim 53. Huppenthal does not specifically disclose a multiplying and summing sequence. However, Hinds discloses an adder circuit configured to receive the first partial result and to generate a non redundant representation of the first partial result and a carry out value. (Hines col. 3, lines 5-17: applying a multiply-accumulate operation to operands; multiplying operands and adding multiplication results; col. 8, lines col. 8, lines 6-25: chained multiply-accumulate operation; carry save adders and usage of partial multiplier (partial results: high order bits); output of results of partial multiplier is normalized and passed to carry save adders and final product adder)
Motivation for Hinds to disclose arithmetic operations is as stated in Claim 1 above.

With Regards to Claim 59, Huppenthal discloses the processor as recited in claim 58. Huppenthal does not specifically disclose a multiplying and summing sequence. However, Hinds discloses the adder circuit is further configured to feed the carry out value back to itself as an input. (Hines col. 3, lines 5-17: applying a multiply-accumulate operation to operands; multiplying operands and adding multiplication results; col. 8, lines col. 8, lines 6-25: chained multiply-accumulate operation; carry save adders and usage of partial multiplier (partial results: high order bits); output of results of partial multiplier is normalized and passed to carry save adders and final product adder) Motivation for Hinds to disclose arithmetic operations is as stated in Claim 1 above.

With Regards to Claim 60, Huppenthal discloses the processor as recited in claim 58. Huppenthal does not specifically disclose a multiplying and summing sequence. However, Hinds discloses the adder circuit is further configured to feed the carry out value back to the second arithmetic structures. (Hines col. 3, lines 5-17: applying a multiply-accumulate operation to operands; multiplying operands and adding multiplication results; col. 8, lines col. 8, lines 6-25: chained multiply-accumulate operation; carry save adders and usage of partial multiplier (partial results: high order bits); output of results of partial multiplier is normalized and passed to carry save adders and final product adder)
Motivation for Hinds to disclose arithmetic operations is as stated in Claim 1 above.

With Regards to Claim 62, Huppenthal discloses the processor as recited in claim 53, Huppenthal does not specifically disclose a multiplying and summing sequence. However, Hinds discloses at least one of the first and second arithmetic structures comprises carry save adder tree columns. (Hines col. 3, lines 5-17: applying a multiply-accumulate operation to operands; multiplying operands and adding multiplication results; col. 8, lines col. 8, lines 6-25: chained multiply-accumulate operation; carry save adders and usage of partial multiplier (partial results: high order bits); output of results of partial multiplier is normalized and passed to carry save adders and final product adder) Motivation for Hinds to disclose arithmetic operations is as stated in Claim 1 above.

With Regards to Claim 66, Huppenthal discloses an apparatus.
Huppenthal does not specifically disclose a multiplying and summing sequence. However, Hinds discloses means for feeding back high order bits of an executed arithmetic instruction, generated by a first arithmetic circuit, to a second arithmetic circuit generating low order bits of a currently executing arithmetic instruction; means for using the second arithmetic circuit to generate a first partial result of the currently executing arithmetic instruction, the first partial result representing the high order bits of the executed arithmetic instruction that are summed with low order bits of a multiplication result of a first number multiplied by a second number; means for using the first partial result in a subsequent computation in the public-key cryptography application. (Hines col. 3, lines 5-17: applying a multiply-accumulate operation to operands; multiplying operands and adding multiplication results; col. 8, lines col. 8,

lines 6-25: chained multiply-accumulate operation; carry save adders and usage of partial multiplier (partial results: high order bits); output of results of partial multiplier is normalized and passed to carry save adders and final product adder)
Motivation for Hinds to disclose arithmetic operations is as stated in Claim 1 above.

Huppenthal does not specifically disclose supporting a public-key cryptography application.
However, Chen discloses supporting a public-key cryptography application (see Chen col. 6, lines 23-25: arithmetic operations to support acceleration of cryptographic functions)
Motivation for Chen to disclose supporting a public-key cryptography application is as stated in Claim 1 above.

With Regards to Claim 67, Huppenthal discloses an apparatus.
Huppenthal does not specifically disclose a multiplying and summing sequence.
However, Hinds discloses means for feeding back high order bits of an executed arithmetic instruction, from a first arithmetic circuit that generated the high order bits, to a second arithmetic circuit generating low order bits of a currently executing arithmetic instruction; means for supplying a third number to the second arithmetic circuit; and means for using the second arithmetic circuit to generate a first partial result, the first partial result being a representation of the high order bits of the executed arithmetic instruction summed with low order bits of a result of a first number multiplied by a second number and with the third number; and means for using the first partial result in

a subsequent computation in the public-key cryptography application. (Hines col. 3, lines 5-17: applying a multiply-accumulate operation to operands; multiplying operands and adding multiplication results; col. 8, lines col. 8, lines 6-25: chained multiply-accumulate operation; carry save adders and usage of partial multiplier (partial results: high order bits); output of results of partial multiplier is normalized and passed to carry save adders and final product adder)

Huppenthal-Hinds does not specifically disclose supporting a public-key cryptography application.
However, Chen discloses configured to support a public-key cryptography application (see Chen col. 6, lines 23-25: arithmetic operations to support acceleration of cryptographic functions)
Motivation for Chen to disclose supporting a public-key cryptography application is as stated in Claim 1 above.

6. Claims **2, 3, 15 - 18, 27 - 29, 35, 43 - 46** are rejected under 35 U.S.C. 103(a) as being unpatentable over **Huppenthal-Hinds-Chen** and further in view of **Lasher et al.** (US Patent No. **4,863,247**).

With Regards to Claim 2, Huppenthal discloses the method as recited in claim 1.
Hinds discloses the high order bits are fed back as stated in Claim 1 above.
Huppenthal does not specifically disclose redundant number representation.
However, Lasher discloses wherein the result is in redundant number representation.

(see Lasher col. 6, lines 6-9; col. 6, lines 16-18: redundant number representations)

It would have been obvious to one of ordinary skill in the art to modify Chen for a result in redundant number representation as taught by Lasher. One of ordinary skill in the art would have been motivated to employ the teachings of Lasher in order that fully parallel carry-free operation is provided for with reduced complexity. (see Lasher col. 2, lines 57-62)

With Regards to Claim 3, Huppenthal discloses the method as recited in claim 2.
Hinds discloses sum and carry bits as stated in Claim 1 above.
Huppenthal does not specifically disclose redundant number representation.
However, Lasher discloses wherein the result is in redundant number representation.
(see Lasher col. 6, lines 6-9; col. 6, lines 16-18: redundant number representations)
Motivation for Lasher to disclose redundant number representation is as stated in Claim 2 above.

With Regards to Claim 15, Huppenthal discloses the method as recited in claim 1.
Hinds discloses the first partial result as stated in Claim 1 above.
Huppenthal does not specifically disclose redundant number representation.
However, Lasher discloses wherein the result is in redundant number representation.
(see Lasher col. 6, lines 6-9; col. 6, lines 16-18: redundant number representations)
Motivation for Lasher to disclose redundant number representation is as stated in Claim 2 above.

With Regards to Claim 16, Huppenthal discloses the method as recited in claim 15. Hinds discloses supplying the first partial result to an adder circuit to generate the first partial result and a carry out value as stated in Claim 1 above.
Huppenthal does not specifically disclose redundant number representation. However, Lasher discloses wherein the result is a non redundant representation. (see Lasher col. 6, lines 6-9; col. 6, lines 16-18: redundant number representations)
Motivation for Lasher to disclose redundant number representation is as stated in Claim 2 above.

With Regards to Claim 17, Huppenthal discloses the method as recited in claim 16. Hinds discloses feeding back the carry out value to the adder circuit as stated in Claim 1 above.

With Regards to Claim 18, Huppenthal discloses the method as recited in claim 16. Hinds discloses feeding back the carry out value to the second arithmetic circuit as stated in Claim 1 above.

With Regards to Claim 27, Huppenthal discloses the method as recited in claim 21. Hinds discloses supplying the first partial result to an adder circuit to generate a non redundant representation of the first partial result and a carry out value as stated in Claim 1 above.

Huppenthal does not specifically disclose redundant number representation. However, Lasher discloses wherein the result is a non redundant number representation. (see Lasher col. 6, lines 6-9; col. 6, lines 16-18: redundant number representations)

Motivation for Lasher to disclose redundant number representation is as stated in Claim 2 above.

With Regards to Claim 28, Huppenthal discloses the method as recited in claim 27. Hinds discloses feeding back the carry out value to the adder circuit as stated in Claim 1 above.

With Regards to Claim 29, Huppenthal discloses the method as recited in claim 27. Hinds discloses feeding back the carry out value to the second arithmetic structures as stated in Claim 1 above.

With Regards to Claim 35, Huppenthal discloses the method as recited in claim 21 wherein the high order bits. (see Chen col. 11, lines 34-40: feedback; first using circuit; then using circuit again with register provided with output from first operational stage; col. 10, lines 13-26: multiple-accumulate instruction; first addend comes from the rightmost k bits of Z register; bits are added to the k bits in the rightmost portion of the product A,B) And, Lasher discloses wherein the result is in redundant number representation. (see Lasher col. 6, lines 6-9; col. 6, lines 16-18: redundant number

representations)

Motivation for Lasher to disclose redundant number representation is as stated in Claim 2 above.

With Regards to Claim 43, Huppenthal discloses the processor as recited in claim 38, Hinds discloses the first partial result as stated in Claim 1 above.
Lasher discloses the result is in redundant number representation as stated in Claim 2 above.

With Regards to Claim 44, Huppenthal discloses the processor as recited in claim 43. Hinds discloses an adder circuit configured to receive the first partial result and to generate a non redundant representation of the first partial result and a carry out value as stated in Claim 1 above.

With Regards to Claim 45, Huppenthal discloses the processor as recited in claim 44. Hinds discloses an adder circuit configured to feed the carry out value back to itself as an input as stated in Claim 1 above.

With Regards to Claim 46, Huppenthal discloses the processor as recited in claim 44. Hinds discloses an adder circuit configured to feed the carry out value back to the second arithmetic structures as stated in Claim 1 above.

7. Claims **11, 20, 30, 31, 37, 47, 61** are rejected under 35 U.S.C. 103(a) as being unpatentable over **Huppenthal-Hinds-Chen** and further in view of **Stribaek et al.** (US Patent No. **7,181,484**).

With Regards to Claim 11, Huppenthal discloses the method as recited in claim 1. Hinds discloses at least one of the first and second pluralities of arithmetic structures as stated in Claim 1 above.
Huppenthal does not specifically disclose whereby a plurality of Wallace tree columns. However, Stribaek discloses wherein further comprises a plurality of Wallace tree columns. (see Stribaek col. 9, lines 10-24; col. 9, lines 37-39: Wallace tree)

It would have been obvious to one of ordinary skill in the art to modify Chen for usage of Wallace tree multiplication as taught by Stribaek. One of ordinary skill in the art would have been motivated to employ the teachings of Stribaek in order to enable the capability for extended precision in arithmetic calculations due to extensive and increasing usage of public key cryptography. (see Stribaek col. 1, lines 61-67)

With Regards to Claim 20, Huppenthal discloses the method as recited claim 1. Hinds discloses storing the high order bits as stated in Claim 1 above.
Huppenthal does not specifically disclose whereby an extended carry register. However, Stribaek discloses wherein an extended carry register. (see Stribaek col. 9, lines 10-24; col. 9, lines 37-39: Wallace tree; col. 5, lines 41-45: extended carry operations)

It would have been obvious to one of ordinary skill in the art to modify Chen for usage of extended carry operations as taught by Stribaek. One of ordinary skill in the art would have been motivated to employ the teachings of Stribaek in order to enable the capability for extended precision in arithmetic calculations due to extensive and increasing usage of public key cryptography. (see Stribaek col. 1, lines 61-67)

With Regards to Claim 30, Huppenthal discloses the method as recited in claim 21. Hinds discloses at least one of the first and second pluralities of arithmetic structures as stated in Claim 1 above.

Huppenthal does not specifically disclose a plurality of Wallace tree columns. However, Stribaek discloses wherein further comprises a plurality of Wallace tree columns. (see Stribaek col. 9, lines 10-24; col. 9, lines 37-39: Wallace tree; col. 2, line 66 - col. 3, line 6: public key cryptographic calculations)

It would have been obvious to one of ordinary skill in the art to modify Chen for usage of Wallace tree multiplication as taught by Stribaek. One of ordinary skill in the art would have been motivated to employ the teachings of Stribaek in order to enable the capability for extended precision in arithmetic calculations due to extensive and increasing usage of public key cryptography. (see Stribaek col. 1, lines 61-67)

With Regards to Claim 31, Huppenthal discloses the method as recited in claim 21. Hinds discloses at least one of the first and second pluralities of arithmetic structures as stated in Claim 1 above.

Huppenthal does not specifically disclose carry save adder tree columns. However, Stribaek discloses wherein further comprises a plurality of adder tree columns. (see Stribaek col. 9, lines 10-24; col. 9, lines 37-39: Wallace tree; col. 5, lines 41-45: extended carry operations; col. 7, lines 31-37; col. 9, lines 10-14: carry-save adder; col. 2, line 66 - col. 3, line 6: public key cryptographic calculations)

It would have been obvious to one of ordinary skill in the art to modify Chen for usage of Wallace tree multiplication as taught by Stribaek. One of ordinary skill in the art would have been motivated to employ the teachings of Stribaek in order to enable the capability for extended precision in arithmetic calculations due to extensive and increasing usage of public key cryptography. (see Stribaek col. 1, lines 61-67)

With Regards to Claim 37, Huppenthal discloses the method as recited in claim 21. Hinds discloses storing the high order bits as stated in Claim 1 above.
Huppenthal does not specifically disclose whereby an extended carry register. However, Stribaek discloses wherein an extended carry register. (see Stribaek col. 9, lines 10-24; col. 9, lines 37-39: Wallace tree; col. 5, lines 41-45: extended carry operations)

It would have been obvious to one of ordinary skill in the art to modify Chen for usage of extended carry operations (register) as taught by Stribaek. One of ordinary skill in the art would have been motivated to employ the teachings of Stribaek in order to enable the capability for extended precision in arithmetic calculations due to extensive and increasing usage of public key cryptography. (see Stribaek col. 1, lines 61-67)

With Regards to Claim 47, Huppenthal discloses the processor as recited in claim 38. Hinds discloses at least one of the first and second pluralities of the arithmetic structures as stated in Claim 1 above.

Huppenthal does not specifically disclose whereby a plurality of Wallace tree columns. However, Stribaek discloses wherein further comprises a plurality of Wallace tree columns. (see Stribaek col. 9, lines 10-24; col. 9, lines 37-39: Wallace tree)

It would have been obvious to one of ordinary skill in the art to modify Chen for usage of Wallace tree multiplication as taught by Stribaek. One of ordinary skill in the art would have been motivated to employ the teachings of Stribaek in order to enable the capability for extended precision in arithmetic calculations due to extensive and increasing usage of public key cryptography. (see Stribaek col. 1, lines 61-67)

With Regards to Claim 61, Huppenthal discloses the processor as recited in claim 53. Hinds discloses at least one of the first and second arithmetic structures as stated in Claim 1 above.

Huppenthal does not specifically disclose whereby further comprising Wallace tree columns.

However, Stribaek discloses wherein further comprises a Wallace tree column. (see Stribaek col. 9, lines 10-24; col. 9, lines 37-39: Wallace tree)

It would have been obvious to one of ordinary skill in the art to modify Chen as taught by Stribaek for usage of Wallace tree multiplication. One of ordinary skill in the

art would have been motivated to employ the teachings of Stribaek in order to enable the capability for extended precision in arithmetic calculations due to extensive and increasing usage of public key cryptography. (see Stribaek col. 1, lines 61-67)

8. Claims **12 - 14, 32 - 34, 49 - 51, 63 - 65** are rejected under 35 U.S.C. 103(a) as being unpatentable over **Huppenthal-Hinds-Chen** and further in view of **Chen et al.** (US Patent No. **6,687,725**: referred to as "Chen2").

With Regards to Claim 12, Huppenthal discloses the method as recited in claim 1
Hinds discloses at least one of the first and second pluralities of arithmetic structures is usable to perform integer multiplication as stated in Claim 1 above
Huppenthal does not specifically disclose XOR operations.
However, Chen2 discloses wherein to perform XOR multiplication. (see Chen2 col. 4, line 64 - col. 5, line 2; col. 15, lines 29-31: XOR operations)

It would have been obvious to one of ordinary skill in the art to modify Chen to perform XOR multiplication as taught by Chen2. One of ordinary skill in the art would have been motivated to employ the teachings of Chen2 in order to provide an arithmetic circuit which can perform all arithmetic operations in the finite field, including addition, multiplication, division, exponentiation and inverse multiplication. (see Chen2 col. 3, lines 17-21)

With Regards to Claim 13, Huppenthal discloses the method as recited in claim 12.

Hinds discloses a logical circuit in at least one of the first and second arithmetic circuits supplying a variable value for integer multiplication mode that varies according to inputs supplied to the logical circuit if in integer multiplication mode, to thereby ensure a result unaffected by carry logic performing carries in integer multiplication mode as stated in Claim 1 above.

Huppenthal does not specifically disclose XOR operations.

However, Chen2 discloses wherein supplying a fixed value if in XOR multiplication mode and to thereby ensure a result is determined in XOR multiplication. (see Chen2 col. 4, line 64 - col. 5, line 2; col. 15, lines 29-31: XOR operations)

It would have been obvious to one of ordinary skill in the art to modify Chen to support XOR operations as taught by Chen2. One of ordinary skill in the art would have been motivated to employ the teachings of Chen2 in order to provide an arithmetic circuit which can perform all arithmetic operations in the finite field, including addition, multiplication, division, exponentiation and inverse multiplication. (see Chen2 col. 3, lines 17-21)

With Regards to Claim 14, Huppenthal discloses the method as recited in claim 13.

Hinds discloses the logical circuit operates as a majority circuit in integer multiplication mode as stated in Claim 1 above.

Huppenthal does not specifically disclose XOR operations.

However, Chen2 discloses wherein outputs a zero in the XOR multiplication mode. (see Chen2 col. 4, line 64 - col. 5, line 2; col. 15, lines 29-31: XOR operations)

It would have been obvious to one of ordinary skill in the art to modify Chen to support XOR operations as taught by Chen2. One of ordinary skill in the art would have been motivated to employ the teachings of Chen2 in order to provide an arithmetic circuit which can perform all arithmetic operations in the finite field, including addition, multiplication, division, exponentiation and inverse multiplication. (see Chen2 col. 3, lines 17-21)

With Regards to Claim 32, Huppenthal discloses the method as recited in claim 21. Hinds discloses at least one of the first and second pluralities of arithmetic structures is usable to perform integer multiplication as stated in Claim 1 above.
Huppenthal does not specifically disclose XOR operations.
However, Chen2 discloses wherein perform both integer and XOR multiplication. (see Chen2 col. 4, line 64 - col. 5, line 2; col. 15, lines 29-31: XOR operations)

It would have been obvious to one of ordinary skill in the art to modify Chen to perform XOR multiplication as taught by Chen2. One of ordinary skill in the art would have been motivated to employ the teachings of Chen2 in order to provide an arithmetic circuit which can perform all arithmetic operations in the finite field, including addition, multiplication, division, exponentiation and inverse multiplication. (see Chen2 col. 3, lines 17-21)

With Regards to Claim 33, Huppenthal discloses the method as recited in claim 32. Hinds discloses a logic circuit in at least one of the first and second pluralities of

arithmetic structures supplying a variable value that varies according to inputs supplied to the logical circuit if in integer multiplication mode, to thereby ensure a result unaffected by carry logic performing carries in integer multiplication mode as stated in Claim 1 above.

Huppenthal does not specifically disclose XOR operations.

However, Chen2 discloses wherein supplying a fixed value if in XOR multiplication mode. (see Chen2 col. 4, line 64 - col. 5, line 2; col. 15, lines 29-31: XOR operations)

It would have been obvious to one of ordinary skill in the art to modify Chen to support XOR operations as taught by Chen2. One of ordinary skill in the art would have been motivated to employ the teachings of Chen2 in order to provide an arithmetic circuit which can perform all arithmetic operations in the finite field, including addition, multiplication, division, exponentiation and inverse multiplication. (see Chen2 col. 3, lines 17-21)

With Regards to Claim 34, Huppenthal discloses the method as recited in claim 33. Hinds the logic circuit operates as a majority circuit in integer multiplication mode as stated in Claim 1 above.

Huppenthal does not specifically disclose XOR operations.

However, Chen2 discloses wherein outputs a zero in the XOR multiplication mode. (see Chen2 col. 4, line 64 - col. 5, line 2; col. 15, lines 29-31: XOR operations)

It would have been obvious to one of ordinary skill in the art to modify Chen to support XOR operations as taught by Chen2. One of ordinary skill in the art would

have been motivated to employ the teachings of Chen2 in order to provide an arithmetic circuit which can perform all arithmetic operations in the finite field, including addition, multiplication, division, exponentiation and inverse multiplication. (see Chen2 col. 3, lines 17-21)

With Regards to Claim 49, Huppenthal discloses the processor as recited in claim 38. Hinds discloses at least one of the first and second pluralities of arithmetic structures is configured to selectively perform one of integer multiplication according to a control signal as stated in Claim 1 above.

Huppenthal does not specifically disclose XOR operations.

However, Chen2 discloses wherein perform one of integer and XOR multiplication. (see Chen2 col. 4, line 64 - col. 5, line 2; col. 15, lines 29-31: XOR operations)

It would have been obvious to one of ordinary skill in the art to modify Chen to support XOR operations as taught by Chen2. One of ordinary skill in the art would have been motivated to employ the teachings of Chen2 in order to provide an arithmetic circuit which can perform all arithmetic operations in the finite field, including addition, multiplication, division, exponentiation and inverse multiplication. (see Chen2 col. 3, lines 17-21)

With Regards to Claim 50, Huppenthal discloses the processor as recited in claim 49. Hinds discloses a plurality of logic circuits in the first and second pluralities of arithmetic structures, each logic circuit responsive to the control signal to supply a variable output

value in integer multiplication mode, the variable output value varying according to values of inputs supplied to the logic circuit, to thereby ensure a result unaffected by carry logic generating carries in integer multiplication mode as stated in Claim 1 above.
Huppenthal does not specifically disclose XOR operations.
However, Chen2 discloses wherein to support XOR operations for binary polynomial fields. (see Chen2 col. 4, line 64 - col. 5, line 2; col. 15, lines 29-31: XOR operations)

It would have been obvious to one of ordinary skill in the art to modify Chen to supply a fixed output value in XOR multiplication mode and ensure a result is determined in XOR multiplication mode as taught by Chen2. One of ordinary skill in the art would have been motivated to employ the teachings of Chen2 in order to provide an arithmetic circuit which can perform all arithmetic operations in the finite field, including addition, multiplication, division, exponentiation and inverse multiplication. (see Chen2 col. 3, lines 17-21)

With Regards to Claim 51, Huppenthal discloses the processor as recited in claim 50. Hinds discloses the logical circuit is configured to operate as a majority circuit in integer multiplication mode as stated in Claim 1 above.
Huppenthal does not specifically disclose XOR operations.
However, Chen2 discloses wherein to output a zero in XOR multiplication mode. (see Chen2 col. 4, line 64 - col. 5, line 2; col. 15, lines 29-31: XOR operations)

It would have been obvious to one of ordinary skill in the art to modify Chen to output a zero in XOR multiplication mode as taught by Chen2. One of ordinary skill in

the art would have been motivated to employ the teachings of Chen2 in order to provide an arithmetic circuit which can perform all arithmetic operations in the finite field, including addition, multiplication, division, exponentiation and inverse multiplication. (see Chen2 col. 3, lines 17-21)

With Regards to Claim 63, Huppenthal discloses the processor as recited in claim 53. Hinds discloses the arithmetic structures are configured to selectively perform one of integer multiplication according to a control signal as stated in Claim 1 above. Huppenthal does not specifically disclose XOR operations.
However, Chen2 discloses wherein to perform XOR multiplication. (see Chen2 col. 4, line 64 - col. 5, line 2; col. 15, lines 29-31: XOR operations)

It would have been obvious to one of ordinary skill in the art to modify Chen to perform XOR multiplication as taught by Chen2. One of ordinary skill in the art would have been motivated to employ the teachings of Chen2 in order to provide an arithmetic circuit which can perform all arithmetic operations in the finite field, including addition, multiplication, division, exponentiation and inverse multiplication. (see Chen2 col. 3, lines 17-21)

With Regards to Claim 64, Huppenthal discloses the processor as recited in claim 63. Hinds discloses a plurality of logic circuits in at least one of the first and second pluralities of arithmetic structures, each logic circuit responsive to the control signal to supply a variable output value in integer multiplication mode, the variable output value

varying according to values of inputs supplied to the logic circuit, to thereby ensure a result is unaffected by carry logic generating carries in integer multiplication mode as stated in Claim 1 above.

Huppenthal does not specifically disclose XOR operations.

However, Chen2 discloses wherein to supply a fixed output value in XOR multiplication mode and to thereby ensure a result is determined in XOR multiplication mode. (see Chen2 col. 4, line 64 - col. 5, line 2; col. 15, lines 29-31: XOR operations)

It would have been obvious to one of ordinary skill in the art to modify Chen to support XOR operations as taught by Chen2. One of ordinary skill in the art would have been motivated to employ the teachings of Chen2 in order to provide an arithmetic circuit which can perform all arithmetic operations in the finite field, including addition, multiplication, division, exponentiation and inverse multiplication. (see Chen2 col. 3, lines 17-21)

With Regards to Claim 65, Huppenthal discloses the processor as recited in claim 64. Hinds discloses the logical circuit is configured to operate as a majority circuit in integer multiplication mode and to output a zero in the XOR multiplication mode as stated in Claim 1 above.

Huppenthal does not specifically disclose XOR operations.

However, Chen2 discloses wherein to output a zero in the XOR multiplication mode. (see Chen2 col. 4, line 64 - col. 5, line 2; col. 15, lines 29-31: XOR operations)

It would have been obvious to one of ordinary skill in the art to modify Chen and to

output a zero in the XOR multiplication mode as taught by Chen2. One of ordinary skill in the art would have been motivated to employ the teachings of Chen2 in order to provide an arithmetic circuit which can perform all arithmetic operations in the finite field, including addition, multiplication, division, exponentiation and inverse multiplication. (see Chen2 col. 3, lines 17-21)

Conclusion

THIS ACTION IS MADE FINAL. Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the mailing date of this final action.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Carlton V. Johnson whose telephone number is 571-270-1032. The examiner can normally be reached on Monday thru Friday , 8:00 - 5:00PM EST.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's

supervisor, Nasser Moazzami can be reached on 571-272-4195. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

Carlton V. Johnson
Examiner
Art Unit 2436

CVJ
July 6, 2010

/David García Cervetti/

Primary Examiner, Art Unit 2436